TEC | Tecnológico de Costa Rica

Programa del curso IC-6831

Aseguramiento de la Calidad del Software

Escuela de Computación
Carrera de Ingeniería de Computación, Plan 410.

I parte: Aspectos relativos al plan de estudios

1 Datos generales

Nombre del curso:	Aseguramiento de la Calidad del Software.
Código:	IC-6831
Tipo de curso:	Teórico - Práctico
Nº de créditos:	3
Nº horas de clase por semana:	4
Nº horas extraclase por semana:	6
Ubicación en el plan de estudios:	Curso del 6° semestre de la carrera de Ingeniería en Computación
Requisitos:	IC-6821 Diseño de Software IC-4810 Administración de Proyectos.
Correquisitos:	Ninguno
El curso es requisito de:	IC-7841 Proyecto de Ingeniería del Software
Asistencia:	Obligatoria
Suficiencia:	No.
Posibilidad de reconocimiento:	Sí.
Vigencia del programa:	I semestre 2015.

2 Descripción general

La evaluación de la calidad del software vía verificación y validación puede ahorrar a una organización esfuerzo, tiempo y dinero. Las mejoras en la calidad del software implican usualmente mejoras en la productividad también. El curso presenta una perspectiva pragmática del control de la calidad del software enmarcado en procesos de ingeniería del software, con énfasis en los procesos y actividades del aseguramiento de la calidad.

3 Objetivos

Objetivo General

Analizar el concepto de la calidad, principios de estándares y técnicas aplicables a la validación y la verificación, desarrollo y mantenimiento de sistemas intensivos en software para la producción de software con mejores características de calidad y costos menores.

Objetivos Específicos

- Explicar el contexto del aseguramiento de la calidad, calidad del software y sus múltiples facetas y atributos.
- Entender el alcance de la calidad en los proyectos de desarrollo y mantenimiento de software.
- Distinguir conceptos de verificación y validación de software, su papel complementario a las actividades de desarrollo en los procesos de software.
- Aplicar métodos, técnicas y estándares de aseguramiento, verificación y validación de calidad a productos de software y procesos de aseguramiento de la calidad.
- Explicar la relación de los procesos de la administración de la configuración del software con los de la administración de la calidad del software.
- Conocer algunas mediciones de calidad que pueden realizarse sobre los productos y los procesos de software.

4 Contenidos

Calidad del software

- Calidad del software. Fracasos de software.
- Requerimientos y calidad.
- Calidad y factores económicos. Costo de la calidad. Costo de la no-calidad.
- Atributos de calidad.
- Aseguramiento de la calidad.
- Control de la calidad.
- Responsabilidades en torno a la calidad.
- Beneficios de aplicar el aseguramiento de la calidad.

Control de la calidad y de sus costos

- Comprobación de características de calidad.
- Defectos del software y terminología asociada.
- Clasificación de defectos.
- Estándar IEEE 1044-1993 (R2002): Clasificación estándar para anomalías de software.
- Costos de detección y de reparación de defectos.
- Verificación y validación.
- Distribución de los defectos en las actividades de desarrollo.
- Prevención de defectos. Detección temprana de defectos. Contención de defectos.

Verificación de software: revisiones estáticas

- Revisiones estáticas: Conceptos y principios generales.
- Modelos en los procesos de software. Revisión de modelos.
- Variedades de revisión estática. Estándar IEEE-1028: Revisiones de software.
- Utilización de listas de cotejo. Ejemplos de listas de cotejo.
- Aplicabilidad de las técnicas según los productos del proceso.
- Disposición de productos.
- Cuantificación de costos y de tiempos.
- Registro y reporte de datos de verificación. Seguimiento.
- Procesos de verificación. Organización y participación en la verificación.
- Relación con los procesos de desarrollo y de mantenimiento.

Validación de software mediante pruebas

- Pruebas de software: conceptos y principios generales.
- Limitaciones de las pruebas.
- Necesidad de objetividad y sistematicidad.
- Tipos de pruebas. Niveles de pruebas. Estrategias de pruebas.
- Técnicas de pruebas funcionales, basadas en especificaciones ("caja negra").
- Técnicas de pruebas funcionales, basadas en estructura ("caja blanca").
- Pruebas exploratorias.
- Aplicabilidad de las técnicas de prueba.
- Cobertura de las pruebas.
- Pruebas de regresión.
- Pruebas técnicas: pruebas de carga, pruebas de tensión, pruebas de recuperación, pruebas de seguridad, pruebas de aplicaciones Web, pruebas de software orientado a objetos, pruebas de software móvil, pruebas de software empotrado, pruebas de usabilidad.
- Especificación de pruebas: diseños, casos, procedimientos. Estándar IEEE 1008-1997 (R2003): Pruebas de unidades de software. Estándar IEEE 829-1998: Documentación de pruebas de software.
- Planes de pruebas.
- Ejecución de las pruebas.
  Registro y reporte de datos de validación. Seguimiento. Relación con depuración, administración de configuración e integración.
- Automatización de las pruebas. Aplicación de herramientas para pruebas.
- Proceso de integración, pruebas de integración y de subsistemas.
- Obtención de criterios de aceptación. Especificación de pruebas de aceptación. Establecimiento de participaciones y responsabilidades. Planes, políticas y procedimientos de pruebas de aceptación.
- Procesos de validación. Organización y participación en la validación.

- Relación con los procesos de desarrollo y de mantenimiento. Retroalimentación del proceso administrativo. Retroalimentación del proceso técnico.

Verificación formal del software

- ¿Qué es verificar formalmente un programa? ¿Cómo nos ayudan la lógica formal y la matemática discreta?
- Propiedades de un lenguaje imperativo ejemplar: semántica axiomática ("lógica de Hoare").
- Anotación y especificación de programas con pre-condiciones, post-condiciones. Variantes e invariantes. Propiedades de los comandos.
- Demostraciones de correctitud mediante lógica de Hoare.
- Desarrollo de programas correctos a partir de especificaciones. Refinamiento algorítmico. Refinamiento de datos.
- Análisis de programas con herramientas lógicas: demostradores de teoremas, comprobadores de modelos.
- Métodos formales en el desarrollo de sistemas de seguridad crítica.

Integración de la verificación y la validación a los procesos de software

- Sinergia de revisiones y pruebas.
- Procesos de desarrollo: lineales, iterativos, ágiles.
- Ciclo V y principios de integración de V&V con procesos de software.
- Técnicas recomendadas por producto.
- Tareas de aseguramiento durante el proyecto.
- Papeles en los procesos de verificación y validación. Participación de los actores durante el proyecto.
- Análisis de modos de falla.
- Valoración de riesgos para la verificación y la validación.

Aseguramiento de la calidad del software

- Actividades de gestión de la calidad en los proyectos informáticos.
- Integración del aseguramiento, la verificación y la validación

con el proceso de software.
- Planificación del aseguramiento de la calidad. Estándar IEEE 730-2002: Planes de aseguramiento de la calidad. El plan de calidad en un proyecto de software.
- Planificación de la verificación y la validación. Estándar IEEE 1012-1998: Verificación y validación de software.
- Agilidad: desarrollo dirigido por pruebas ("test-driven development").
- Estimación del esfuerzo dedicado a la verificación y la validación.
- Integración del plan de calidad con el plan general del proyecto.
- La función de aseguramiento de la calidad. Auditoría de procesos, de productos y de proyectos.
- Problemas de calidad: identificación, análisis, seguimiento, reporte, resolución de incidentes. Análisis de causas raíz.
- Contención de defectos y su medición. Medición de la calidad.
- Alineamiento del aseguramiento de la calidad del software con el sistema organizacional de calidad.

Administración de configuraciones de software
- Configuraciones de software: ítemes y administración.
- Actividades y organización de la administración de la configuración del software.
- Control de cambios en el software. Herramientas para manejo de versiones.
- Liberación de versiones de configuraciones de software.
- Auditoría de configuraciones de software.
- Relaciones entre el aseguramiento y el control de la calidad, la administración de la configuración y el desarrollo/mantenimiento de software.

II parte: Aspectos operativos

5 Metodología de enseñanza y aprendizaje

El curso se impartirá mediante clases magistrales dirigidas por el profesor. Para mejorar la comprensión y poner en práctica algunos conceptos, se realizarán actividades grupales y discusión de casos durante las clases.

En la mayoría de las clases se asignarán artículos o capítulos de libros y usualmente se harán evaluaciones, discusiones o comprobaciones de lectura sobre dichas lecturas.

Adicionalmente, los estudiantes deberán elaborar un proyecto práctico donde construyen un producto de software de mediano alcance, a la vez que realicen o simulen actividades de calidad y pruebas en el desarrollo de dicho producto.

Con base en los requerimientos, restricciones, herramientas y pautas dadas por el profesor, el grupo de trabajo debe elaborar una serie de productos que se entregarán paulatinamente.

Se espera la participación activa de los estudiantes en las lecciones, a través de preguntas e intervenciones que aporten valor a la discusión de los temas.

6 Evaluación

Proyecto	50%
Exámenes parciales	30%
Pruebas cortas, tareas y exposiciones	20%

El proyecto tiene como objetivo que el estudiante resuelva un problema práctico a partir de una guía propuesta por el profesor, de manera que les permita la aplicación y profundización de los conocimientos adquiridos en el curso. El mismo se desarrollará durante todo el semestre y podrán ser realizados en grupos de máximo cuatro personas. Las normas para la documentación, desglose de porcentajes internos de cada entrega y criterios de evaluación se entregarán oportunamente.

Se realizarán dos exámenes parciales a través del semestre, ambos con el mismo peso en la nota final. Los exámenes cubrirán los contenidos del curso y más que una comprobación de lectura o de memorización de contenidos, intentarán determinar si el estudiante asimiló e interiorizó los conocimientos y posee los criterios necesarios para aplicar los temas cubiertos en proyectos de la vida real.

Las pruebas cortas se efectuarán en el momento de la clase que el profesor considere más apropiado y no necesariamente se informarán con previo aviso. Por ningún motivo se realizarán reposiciones de pruebas cortas.

Para el cálculo del porcentaje de pruebas, tareas y exposiciones se corresponderá a calcular la proporción de todas las calificaciones obtenidas, donde cada entregable tendrá un peso idéntico para el cálculo de dicho rubro, excepto en los casos en que se asigne una prueba o exposición cuyo alcance y complejidad sea mayor al de los restantes ítems. Dichos mini-proyectos y exposiciones tendrán un valor doble o triple al de los restantes ítems.

En caso de que se detecte un plagio o intento de fraude en cualquier trabajo, asignación o evaluación por parte de un estudiante, se procederá a anular el mismo y se enviará una carta al expediente del estudiante.

El orden, precisión, concisión, ortografía y redacción será evaluado en todos los documentos presentados. El profesor está en la libertad de revisar o no los documentos que no cumplan con dicho criterio.

Cronograma de actividades de evaluación

	Semana
Primer entregable del proyecto	8
Primer examen parcial	9
Segundo entregable del proyecto	14
Segundo examen parcial	18
Tercer entregable del proyecto	19

7 Bibliografía

- Amman, P. & Offutt, J. Introduction to Software Testing. Cambridge, Reino Unido: Cambridge University Press, 2008.
- Astels, David. Test-driven development: a practical guide. Prentice-Hall, 2003.
- Backhouse, Roland. Program Construction: Calculating Implementations from Specifications. John Wiley & Sons, 2003.
- Beizer, Boris. Software testing techniques. Van Nostrand Reinhold, 1983.
- Beizer, Boris. Software system testing and quality assurance. Van Nostrand Reinhold, 1984.
- Beizer, B. Black-box testing; techniques for functional testing of software and systems. John Wiley & Sons, 1995.
- Black, Rex. Managing the testing process, 2nd. Ed. John Wiley and Sons, 2002.
- Binder, R. V. Testing object-oriented systems: models, patterns and tools. Addison-Wesley, 2000.
- Cardoso, Rodrigo. Verificación y desarrollo de programas. Ediciones Uniandes, 1993.
- Copeland, Lee. A practitioner's guide to software test design. Artech House Publishers, 2004.
- Craig, Rick; Jaskiel, Stefan. Systematic software testing. Artech House Publishers, 2002.
- Daughtrey, T. Fundamental concepts for the software quality engineer. American Society for Quality, 2002.
- Dustin, Elfriede et al. Quality Web systems. Addison-Wesley, 2001.
- Ebenau, R. G.; Strauss, S. H. Software inspection process. McGraw-

Hill; 1994.
- Galin, Daniel. Software quality assurance. Pearson / Addison-Wesley, 2004.
- Gries, David. The science of programming. Springer-Verlag, 1981.
- Hehner, Eric. A practical theory of programming. Springer-Verlag, 1994, 2004. Disponible en http://www.cs.utoronto.ca/~hehner/aPToP/
- Hetzel, B. The complete guide to software testing. QED Information Sciences, 1989.
- IEEE. IEEE Std. 829. IEEE Standard for Software Test Documentation. Nueva York, Estados Unidos: The Institute of Electrical and Electronics Engineers.
- IEEE. IEEE Std. 1044. IEEE Standard Classification for Software Anomalies. Nueva York, Estados Unidos: The Institute of Electrical and Electronics Engineers.
- IEEE. IEEE Std. 1028. IEEE Standard for Software Reviews. Nueva York, Estados Unidos: The Institute of Electrical and Electronics Engineers.
- ISO/IEC. ISO/IEC 9126-1:2001, Software Engineering – Software Product Quality. Ginebra, Suiza: ISO/IEC.
- Jorgensen, P. Software Testing: A Craftsman's Approach (tercera edición). Pennsauken, Estados Unidos: Auerbach, 2008.
- Jackson, Daniel. Software abstractions: logic, language and analysis. MIT Press, 1996.
- Kaldewaij, Anne. Programming: the derivation of algorithms. Prentice-Hall International, 1990.
- Kaner, C.; Falk, J.; Nguyen, H. Q. Testing computer software. John Wiley & Sons, 1999.
- Mathur, A. Foundations of Software Testing. Boston, Estados Unidos: Addison-Wesley, 2008.
- Morgan, Carroll. Programming from specifications, 2nd. Ed. Prentice-Hall International, 1994.
- Nguyen, Hung et al. Testing applications on the Web, 2nd. Ed. Wiley, 2003.
- **Patton, Ron. Software testing, 2nd. Ed. SAMS, 2005.**
- Pol, M.; Teunissen, R.; Veenendaal, E. van. Software testing; a guide to the Tmap approach. Addison-Wesley, 2002.
- Splaine, S.; Jaskiel, S. P. The Web testing handbook. STQE Publishing,

2001.
- Wiegers, K. E. Peer reviews in software; a practical guide. Pearson / Addison-Wesley, 2002.

8 Canales de comunicación

Correo del profesor: cariasinter@gmail.com
Cualquier correo enviado al profesor deberá tener el siguiente formato / estilo:
Asunto: QA – *[asunto],* donde asunto corresponde al tema o motivo de la consulta.
Cuerpo: Realice la consulta de la forma más clara y completa posible, usando un estilo de comunicación formal. Cuide la ortografía. Adjunte capturas de pantallas o cualquier otro material que complemente el correo. Al final indique su nombre completo. Se recomienda leer el correo completo y editar lo que corresponda antes de su envío final.

Si el correo corresponde a una consulta de más de un estudiante, o relacionada con un trabajo grupal, copie en el correo a todos los miembros del grupo e indique el nombre de todos los miembros al final del correo.

El estudiante es también bienvenido en consultas presenciales. Los horarios de consulta son los **viernes de 10:00 am a 12 md y, con cita previa, los viernes de 2 pm a 4 pm**. La consulta presencial se realiza en las oficinas del TEC, segunda planta, sobre el comedor estudiantil. El teléfono de oficina es el 2431-3987

9 Profesor

Carlos Arias Rodríguez. Bachiller en Computación del Instituto Tecnológico de Costa Rica y Master en Computación de la Universidad de Costa Rica. Combina labores de docencia universitaria con la consultoría en el desarrollo y optimización de aplicaciones de Internet.